CLAIMS:

1. An audio encoding scheme for a stream that carries audio and video data, which scheme has a mean effective audio frame length $\overline{F}$ that equals the video frame length $1/f_v$ over an integral number M video frames, by provision of audio frames variable in length F in a defined sequence $F(j)$ at encoding.

2. An encoding scheme according to claim 1 in which the frame length F is adjusted by varying an overlap O between successive audio frames.

3. An encoding scheme according to claim 1 or claim 2 in which the value $F(j)$ repeats periodically on j, the periodicity of $F(j)$ defining a sequence of frames.

4. An encoding scheme according to claim 3 having M video and N audio frames per sequence, each audio frame being composed of k blocks of t samples each.

5. An encoding scheme according to claim 4 in which a total overlap O_T between frames in the sequence is equal to $O_T = p \times O + q \times (O+1)$, where O is an overlap length in blocks where $p \in \aleph \wedge q \in \aleph \wedge O \in \aleph \wedge O_T \in \aleph$.

6. An encoding scheme according to claim 5 in which only audio frames corresponding to a particular video frame are overlapped.

7. An encoding scheme according to claim 6 in which $p = (N-M) \times (O+1) - O_T$ and $q = (N-M) - p$.

8. An encoding scheme according to claim 5 in which only audio frames corresponding to a particular video sequence are overlapped.

9. An encoding scheme according to claim 8 in which $p = (N-1)\times(O+1)-O_T$ and $q = (N-1)-p$.

10. An encoding scheme according to claim 5 in which any adjacent audio frames are overlapped.

11. An encoding scheme according to claim 10 in which $p = N\times(O+1)-O_T$ and $q = N-p$.

12. An encoding scheme according to any one of claims 4 to 11 in which $\exists n \in \aleph^+ : n\times t = M\times\left(\frac{f_A}{f_V}\right)$.

13. An audio encoding scheme for a stream that encodes audio and video data in which scheme audio samples of N quasi video-matched frames are encoded in frames with a semi-variable overlap whereby the effective length of the audio frames coincides with the length of a sequence of M video frames, where M and N are positive integers.

14. A data stream encoded by a scheme according to any preceding claim.

15. A data stream according to claim 14 which includes audio frames, each of which is tagged to indicate the size of the audio frame.

16. A data stream according to claim 14 or claim 15 which includes audio frames, each block of which is tagged to indicate whether or not the block is a redundant block.

17. An audio encoder for coding audio for a stream that carries audio and video data in which the encoder produces audio frames of variable length such that a mean effective audio frame length $\overline{F}$ equals the video frame length $1/f_V$ over an integral number M video and N audio frames, by provision of audio frames variable overlap to have an effective in length F in a defined sequence $F(j)$ at encoding.

18. An audio encoder according to claim 17 for coding a stream having a short overlap of length O and a total of q long overlaps in a sequence, the encoder calculating the head overlap using an algorithm that repeats after N frames.

19. An audio decoder for decoding a stream that encodes audio and video data, which decoder calculates an expected effective frame length of an incoming frame, adjusts the actual length of the incoming frame to make it equal to the expected frame length, determines whether any block within a received frame is a redundant block or a non-redundant block, mapping the non-redundant blocks onto sub-band samples.

20. An audio decoder according to claim 19 which modifies the overlap status of blocks in the data stream by application of one or more of a set of block operators to each block.

21. An audio decoder according to claim 20 in which the set of operators includes one or more of: NOP, an operator that does not change the status of a blocks; DROP, an operator that changes the first non-redundant block from the head overlap into a redundant block; APPEND, an operator that changes the first redundant block from the tail overlap into a non-redundant block; and SHIFT, an operator that is a combination of both DROP and APPEND operators.